

## CSA0961 – JAVA

### JF 4-2 PRACTISE

1. Name the components that comprise a .java file. List the components in the order that you would expect to see them in a Java program.

**1. Package Declaration** (optional): Specifies the package name if the class belongs to a package.

```
package com.example;
```

**2. Import Statements** (optional): Includes necessary classes or entire packages from the Java library or other libraries.

```
import java.util.ArrayList;
```

```
import java.util.List;
```

**3. Class Declaration:** Defines the class and its attributes, methods, and constructors.

```
public class Student {  
    // Class body  
}
```

**4. Class Body:** Contains fields, constructors, methods, and inner classes

```
public class Student {  
    // Fields  
    private String fName;  
    private String lName;  
    private String stuId;  
    private String stuStatus;  
    // Constructor  
    public Student(String fName, String lName, String stuId, String stuStatus) {  
        this.fName = fName;  
        this.lName = lName;  
        this.stuId = stuId;  
        this.stuStatus = stuStatus;  
    }  
    // Methods  
    public void printInfo() {  
        System.out.println("Student Name: " + fName + " " + lName);  
        System.out.println("Student ID: " + stuId);  
        System.out.println("Student Status: " + stuStatus);  
    }  
}
```

2. Describe the difference between upper camel case and lower camel case and provide an example of when you would them.

□ **Upper Camel Case:** Also known as Pascal Case, it capitalizes the first letter of each word, including the first one. It's typically used for class names.

- **Example:** 'StudentRecord', 'EmployeeDetails'

□ **Lower Camel Case:** Also known as camelCase, it capitalizes the first letter of each word except the first one. It's generally used for method names, variable names, and parameters.

- **Example:** 'studentName', 'getStudentId'.

3. What syntax is used to import the entire Java utilities package? And if you import an entire package do you also need to import additional classes in the same package separately?

```
import java.util.*;
```

4. Write the syntax for a simple Java object class named Student with the following format: Student Name: Lisa Palombo Student ID: 123456789 Student Status: Active The student information will be stored in the following variables: fName, lName, stuId, stuStatus.

```
public class Student {  
  
    // Variables  
  
    private String fName;  
  
    private String lName;  
  
    private String stuId;  
  
    private String stuStatus;  
  
  
    // Constructor  
  
    public Student(String fName, String lName, String stuId, String stuStatus) {  
  
        this.fName = fName;  
  
        this.lName = lName;  
  
        this.stuId = stuId;  
  
        this.stuStatus = stuStatus;  
  
    }  
  
  
    // Method to print student information  
  
    public void printInfo() {  
  
        System.out.println("Student Name: " + fName + " " + lName);  
  
        System.out.println("Student ID: " + stuId);  
  
        System.out.println("Student Status: " + stuStatus);  
  
    }  
  
}
```

5. Write the code for a Driver Class that will create a Student Object and print the information about the object to the screen.

```
public class Driver {  
  
    public static void main(String[] args) {
```

```

// Creating a Student object

Student student = new Student("Lisa", "Palombo", "123456789", "Active");


// Printing the student information

student.printInfo();

}

}

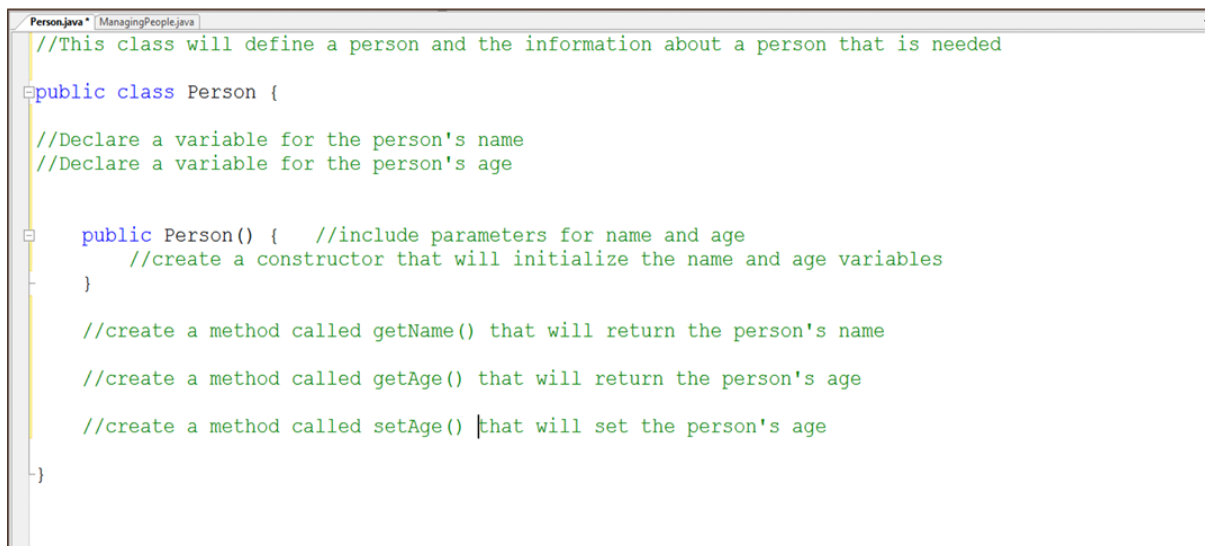
```

6. From this lesson, list 10 Java keywords.

Here are ten commonly used Java keywords:

1. 'class'
2. 'public'
3. 'private'
4. 'protected'
5. 'static'
6. 'void'
7. 'return'
8. 'new'
9. 'if'
10. 'else'

7. Complete the programmer-created object class below. Read the comments for instructions



```

Person.java * [ManagingPeople.java]
//This class will define a person and the information about a person that is needed
public class Person {

    //Declare a variable for the person's name
    //Declare a variable for the person's age

    public Person() {    //include parameters for name and age
        //create a constructor that will initialize the name and age variables
    }

    //create a method called getName() that will return the person's name

    //create a method called getAge() that will return the person's age

    //create a method called setAge() that will set the person's age

}

```

```

package helloworld;
class Person{
    private String name;
    private int age;
    public Person(String name,int age) {
        this.name=name;
        this.age=age;}
    public String getName(){
        return name;}
}

```

```

public void setName(String n){
    this.name=name;}
public int getAge(){
    return age;}
public void setAge(int a){
    this.age=age;}}
public class hellomain{
    public static void main(String[] args){
        Person p=new Person("eliyazar",19);
        System.out.println("name is "+p.getName());
        System.out.println("age is : "+p.getAge());}}

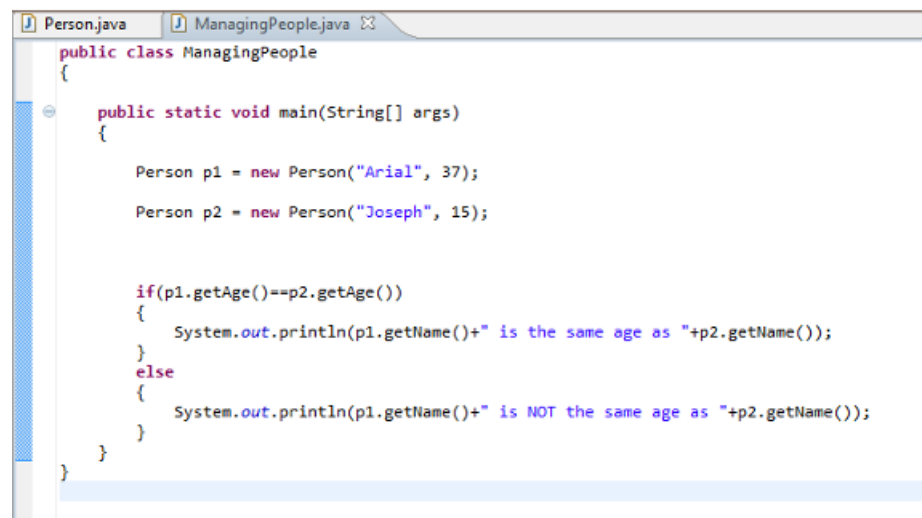
```

```

name is eliyazar
age is : 19

```

8. Use the following driver class to test your results from above.

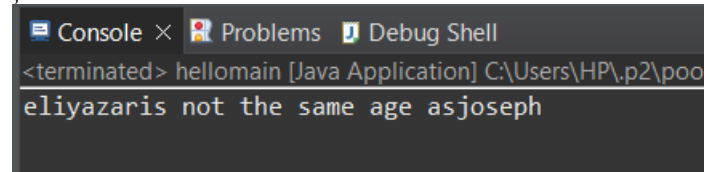


```

package helloworld;
class Person
{
    private String name;
    private int age;
    public Person(String name,int age)
    {
        this.name=name;
        this.age=age;
    }
    public String getName()
    {
        return name;
    }
    public void setName(String name)
    {
        this.name=name;
    }
    public int getAge()
    {
        return age;
    }
    public void setAge(int age)
    {
        this.age=age;
    }
}
public class hellomain
{
    public static void main(String[] args)
    {
        Person p1=new Person("eliyazar",19);
        Person p2=new Person("joseph",15);
    }
}

```

```
        if(p1.getAge()==p2.getAge())
        {
            System.out.println(p1.getName()+"is the same age as"+p2.getName());
        }
        else
        {
            System.out.println(p1.getName()+"is not the same age as"+p2.getName());
        }
    }
}
```



The screenshot shows an IDE console window with three tabs: 'Console', 'Problems', and 'Debug Shell'. The 'Console' tab is active, displaying the output of a Java application. The first line shows the prompt '<terminated>' followed by the command 'hellomain [Java Application] C:\Users\HP\p2\poo'. The second line shows the output 'eliyazaris not the same age asjoseph'.

```
<terminated> hellomain [Java Application] C:\Users\HP\p2\poo
eliyazaris not the same age asjoseph
```